Early last summer, I was a participant in the IEEE History of Electronics Conference at Bletchley Park, the famous site of British code breaking activities during World War II. It is the place where Alan Turing worked, where Tommy Flowers built the Colossus, and a staff of workers used the Bombe to crack the German Enigma code (for the conference report, see *Annals*, vol. 26, no. 3, pp. 62-64).

In perhaps the biggest lesson of the visit, I learned how industrial the process of code breaking really was. Much of Bletchley really acted as a small information processing factory. Each day the staff would receive radio intercepts. They would punch these messages onto paper tapes and put the tapes on the machines. The machines would work through different combinations of letters until they found the key to the cipher. Once the workers had the key, they would go to their decoding equipment and start to translate the messages.

Like the work of code breaking, the practice of history can also be a hard, repetitious task. You can spend hours going through old boxes of records or listening to a noisy interview and get only the smallest bits of usable information. This kind of activity constitutes the bulk of my working days. I sift through piles of evidence, trying to build a coherent picture of an individual, an industry, or a design team. Once in a while, I am rewarded with a moment of excitement. After leaving Bletchley, I spent a little time with the Charles Babbage papers in the British Library. As I leafed through the correspondence, I found things that I had not expected. Babbage corresponded with naturalist Charles Darwin, political theorist Karl Marx, geologist Charles Lyell, poet William Wordsworth, and socialite Crabb Robinson. I had not understood that Babbage had circulated so widely in English society or that he had so many prominent friends.

**In this issue**

In this issue, as in all issues, we have articles that combine the hard work of daily historical labor combined with those all too brief stunning insights into computing history. Perhaps the best example of this is the article on Edmund Berkeley. For many years, I had underestimated Berkeley as a figure in computer science. I knew him as an organizer of the Association for Computing Machinery, an actuary at Prudential Life Insurance, and as the author of the book *Giant Brains*. In some ways, *Giant Brains* summarized my opinion of Berkeley. I viewed the book as an attempt to create some enthusiasm for the new electronic computers. It’s filled with broad claims for the new technology and only ambitious predictions for what these machines might do. That image was changed radically by the work of Bernadette Longo. Longo has worked through the Berkeley papers at the Charles Babbage Institute in Minneapolis and has come to appreciate that Berkeley was something more than a mere promoter of a new idea. He had a vision for transforming society with computers. This vision was based not just on the potential of digital electronic circuits but also upon a strong and deeply felt social philosophy.

Longo’s article, which gives us a new understanding of Berkeley, is matched with a new piece on the computers of the Univac Corporation by George Gray and Ronald Smith. This is the fourth article by these two authors, who have worked hard to surprise us with the early innovations that occurred outside of IBM. In addition, we have two articles by Jack Copeland, one concerning Howard Aiken and the other dealing with the Colossus. In the former, he helps us to understand how clearly Aiken saw the future of computation and in the latter, he has found some detail about the operations of Bletchley Park and the construction of their Colossus proto-computer. Finally, Lee Keet writes about his experiences during the early days of software when he worked at IBM and later founded Turnkey Systems.

We also have a special contribution to the Events and Sightings department that came from my trip to the IEEE History of Electronics Conference. During that meeting, I spoke with Maurice Wilkes, a board member of this journal and a long time supporter. He showed me an email exchange between himself and Allan Bromley, who was one of the great Babbage scholars and a regular contributor to *Annals*. This exchange shows two individuals, one a dedicated scholar and the other a computer pioneer, going over ideas one more time to gain some new insight into that fundamental scientist, Charles Babbage.
Remembering Iverson

As this issue goes to press, we again have to mark the passing of one of the founders of the field, Kenneth Iverson.1 Iverson was best known for the programming language, APL. He invented this language when he was a graduate student at Harvard in 1962. To those who were thinking in terms of the conventional languages of the day—Fortran IV, Cobol, Algol 60—APL seemed like a radical departure. It used a host of symbols to reduce large programs for data manipulation into simple expressions. As with many innovations in computer science, APL had its devoted partisans and its shameless critics. Partisans praised the language because it produced compact and succinct programs. They often pointed to a classic program of 17 symbols that produced a list of prime numbers less than some input value:

\[
\text{Primes: } (\sim R \circ \times R) / R \leftarrow 1 \downarrow R
\]

Yet these same short little codes drew fire from critics who found them incomprehensible and impossible to decode. “There are three things a man must do before his life is done,” wrote the author Stan Kelly-Bootle. “Write two lines in APL and make the buggers run.”2 This controversy lasted well into the 1980s, even though by then APL was recognized as merely an example of a functional language, as compared to the state-oriented languages of Fortran and the rest. I can recall being at an applied mathematics conference in the summer of 1989 and watching two scientists square off to debate the merits of APL, one claiming that the language was the future of scientific computing, the other dismissing APL as an incomprehensible mess.

In hearing the news of Iverson’s passing, I was surprised to be reminded that the inventor of APL was a Canadian. I suppose that this fact exposed a bit of American-centrism in my thinking that overlooks the important contributions to computer science that were done not in Silicon Valley or New York or in Boston, but in Toronto, Calgary, and Waterloo. But such is the lesson of history. There is much that must be done to tell the story of the past but in that work there are at least a few surprises.

References and notes